# 2023년 제1회 게임 프로그래밍 전문가
# 국가기술 자격검정 필기시험문제

| 시험시간 | 2시간 | 형별 | 1-A<br>9매 | 수험번호 | | 성명 | |
|---|---|---|---|---|---|---|---|

**답안카드 작성 시 시험문제지 형별 누락, 수험번호 등 마킹 착오로 인한 불이익은 전적으로 수험자의 귀책사유임을 알려드립니다. 답안카드 뒷면 유의사항을 반드시 확인하시고 답안은 컴퓨터용 수성 사인펜으로만 마킹하십시오.**

## 1 게임 프로그래밍 방법론

1. C++ 언어의 생성자(constructor)에 대한 설명이다. 설명 중 틀린 것은?
① 생성자의 이름은 클래스 명과 같아야 한다.
② 생성자는 반드시 public 멤버함수여야 한다.
③ 클래스의 인스턴스가 생성될 때 호출된다.
④ 리턴타입을 선언할 수 없다.

2. C++언어의 오버라이딩과 오버로딩에 대한 설명이다. 틀린 것은?
① 오버라이딩(overriding)은 부모 클래스에서 정의된 멤버 함수를 재정의 하는 것을 의미한다.
② 오버로딩(overloading)은 같은 이름의 멤버 함수를 여러 개 정의하는 것을 의미한다.
③ 오버라이딩을 위해서는 부모의 멤버 함수와 동일한 함수 이름, 리턴타입, 인자를 가져야 한다.
④ 리턴타입만 달라도 오버로딩이 가능하다.

3. 표준 C언어 컴파일러를 기준으로 할 때, 컴파일 에러가 발생하지 않는 것은?
① long 1Text;
② short Num, int Num2;
③ char _float;
④ int my num;

4. 다음은 열거형을 정의한 코드다.YesterDay의 값은?

```
enum MyEnum {
    YesterDay,
    ToDay = 10,
    Tomorrow };
```

① -1 ② 0 ③ 1 ④ 9

5. 다음은 C++언어로 작성된 코드이다. 코드의 실행결과로 올바른 것은?

```
#include <iostream>
using namespace std;

class Point {
    int x, y;
public:
    Point(int x = 0, int y = 0) : x(x), y(y) {}
    Point(const Point& p) { x = p.x + 1; y = p.y + 1; }
    Point& operator=(const Point& p) {
        if (this != &p) {
            x = p.x - 1;
            y = p.y - 1;
        }
        return *this;
    }
    void print() { cout << "(" << x << ", " << y << ")" << endl; }
};

int main() {
    Point p1(1, 2);
    Point p2(p1);
    Point p3;
    p3 = p1;
    p1.print();
    p2.print();
    p3.print();
    return 0;
}
```

① (1,2) (2,3) (0,1)
② (1,2) (2,3) (3,4)
③ (0,1) (1,2) (2,3)
④ (1,2) (0,1) (2,3)

6. 표준 C언어 컴파일러로 printf()함수에 의해서 출력되는 결과값으로 올바른 것은?

```
#include <stdio.h>
int main(void)
{
    char a=128;
    printf("%d", a );
    return 0;
}
```

① 0 ② -1 ③ -128 ④ 128

7. 다음 코드를 실행했을 때 출력되는 값으로 올바른 것은?

```
#include <stdio.h>
void CallByAddress( int* a, int* b )
{    int* pTemp;
     pTemp = a;
     a = b;
     b = pTemp;

     *pTemp = *b;
}
int main()
{   int nVal1=1, nVal2=10;
    CallByAddress(&nVal1, &nVal2);
    printf("%d,%d ₩n",nVal1, nVal2);
    return 0;
}
```

① 1,1 ② 1,10 ③ 10,1 ④ 10,10

8. 다음은 C++언어에서 난수를 생성하는 간단한 코드다. 그러나 아래 코드는 문제가 있어 보인다. 문제점에 대해서 올바르게 설명한 것은?

```
srand(100);
char  Value = rand()%1000;
printf("%d", Value);
```

① rand()로 만들어지는 난수는 실수이므로 char 변수로 받을 수 없다.
② rand()%1000;을 실행한 결과는 0~1000 중에 하나의 값이다.
③ printf("%d", Value);열의 printf문의 형식 지정자를 %d가 아닌, %f를 사용해야 한다.
④ srand()에 동일한 인자를 넣어 줄 경우, 이후 rand()함수를 호출하면 매번 순서대로 동일한 값이 나온다.

9. 다음은 c언어에서 메모리를 동적으로 할당하는 코드이다. 이 코드는 완전하지 못한데, 어떤 문제가 발생할 수 있는지 제대로 설명한 것은?

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>

int main(void)
{
    char* pText=(char*)malloc(100);
    strcpy(pText, "  aa  " );

    free(pText);
    return 0;
}
```

① strcpy(pText, " aa " ); 코드를 보면 앞쪽과 뒤쪽에 공백문자가 있어서, 공백문자만큼 빠지고 저장된다.
② char* pText=(char*)malloc(100); 이 라인을 보면, 메모리를 char 포인터 형으로 만들었는데, 이것은 잘못된 사용법이다.
③ pText가 메모리를 할당받지 못한 경우에 대한 예외 처리가 없다. 정상적으로 메모리를 할당받지 못하면 pText의 주소값은 NULL이 된다.
④ malloc()함수로 메모리를 할당받았을 경우, free()함수로 메모리 할당을 해제하지 않아도 아무런 문제가 발생하지 않는다.

10. 다음 설명과 일치하는 C언어의 자료형은?

| |
|---|
| - 사용자 정의 자료형이다.<br>- 하나 이상의 변수를 묶어서 그룹화 한다.<br>- 서로 다른 자료형의 멤버들이 동일한 메모리 공간을 공유한다. |

① 구조체 ② 열거형 ③ 포인터 변수 ④ 공용체

11. (ㄱ)에 들어가기에 적절한 것을 다음 중에서 고르시오

| |
|---|
| (ㄱ)는 블록체인에 저장된 데이터 단위로, 고유하면서 상호 교환할 수 없는 토큰을 뜻한다. 대체 가능한 토큰들은 각기 동일한 가치를 가지고 서로 교환이 가능하나 (ㄱ)는 각기 고유성을 지닌다. (ㄱ)는 영구적으로 블록체인에 남기 때문에 고유성을 보장받을 수 있다. (ㄱ)는 근래에 게임과 결합하여 게임 내 아이템이나 캐릭터 등이 하나의 디지털 자산이 되기도 한다. |

① NPT ② NFC ③ NFT ④ NNF

12. 아래의 설명에 부합되는 프로그래밍 언어를 고르시오

| |
|---|
| 모질라 리서치에서 개발한 범용 프로그래밍 언어이다. 인터넷에서 실행되는 서버 및 클라이언트 프로그램을 개발하는데 적합한 언어를 목표로 설계되었다. 이 목표에 따라 안전성과 병행 프로그래밍, 그리고 메모리 관리의 직접 제어에 초점을 맞추고 있다. 또한 성능 면에서는 C++와 비슷한 수준을 목표로 하고 있다.<br><br>메모리 오류를 발생시키지 않도록 설계되었다. 널(NULL) 포인터나 초기화되지 않은 포인터가 존재하지 않도록 강제하고 있다. 모든 변수는 초기값을 가지고 할당되며, 해제된 포인터에 접근하는 코드는 컴파일러가 미리 감지하여 컴파일 오류를 일으킨다. |

① Basic ② Pascal ③ Zig ④ Rust

13. 아래의 설명에 부합되는 직렬화 방식을 고르시오

| |
|---|
| 구글에서 오픈소스로 공개한 구조화된 데이터를 직렬화(Serialization) 하는 방식이다. C++, C#, Java, Python 등 다양한 언어를 지원하고, 직렬화 속도가 빠르고 직렬화된 파일의 크기도 작다. |

① Protocol Buffer ② Fast Buffers ③ Message Pack ④ Bson

14. 아래는 관계형 데이터베이스에 유저 정보가 저장되어 있는 user_table 테이블로부터 level 컬럼의 값을 기준으로 level 이 큰 순서대로 정렬하여(내림차순으로) 유저 정보들만 가져오는 쿼리이다. (ㄱ) 과 (ㄴ) 에 들어가야 하는 것으로 올바르게 짝지워진 것을 고르시오

```
SELECT * FROM user_table (ㄱ) level (ㄴ)
```

① (ㄱ) : NUMBER, (ㄴ) : ASC
② (ㄱ) : ORDER BY, (ㄴ) : DESC
③ (ㄱ) : ORDER BY, (ㄴ) : ASC
④ (ㄱ) : COUNT, (ㄴ) : DESC

15. 아래의 스택 메모리(Stack Memory)에 대한 설명 중 틀린 것을 고르시오
① 함수 안에 선언되는 지역 변수, 매개 변수가 저장된다
② 힙(Heap)에 비해 빠르다
③ 스택 크기에 제한이 있다
④ FIFO(선입 선출) 방식으로 동작한다

16. 아래의 메모리 장치 중 가장 접근(Access) 시간이 빠른 것을 고르시오
① 캐시(L1, L2, L3) ② 하드 디스크(HDD, SSD)
③ 레지스터 ④ 메모리(RAM)

17. 아래의 정적 라이브러리(Static Library)에 대한 설명 중 틀린 것을 고르시오
① 프로그램 빌드 시에 라이브러리가 제공하는 코드를 실행 파일에 넣는 방식의 라이브러리를 의미한다
② 컴파일의 링크 단계에서 실행 파일에 결합된다
③ 실행 파일에 다 들어가기 때문에 실행 시 라이브러리가 필요없다
④ 같은 정적 라이브러리를 사용한 여러 프로그램을 실행할 경우 메모리를 절약한다

18. 아래의 스레드(thread)에 대한 설명 중 맞는 것을 고르시오
① 프로세스 컨텍스트 스위칭에 비해 오버헤드가 작다
② 컨텍스트 스위칭을 자주 해야 CPU를 중단 없이 사용할 수 있으므로 더 좋다
③ 컴퓨터의 CPU 코어 수 보다 더 많은 스레드는 생성할 수 없다
④ 스레드를 많이 만들면 만들수록 CPU를 더 많이 사용할 수 있다

19. 아래의 UDP 설명 중 틀린 것을 고르시오
① 전송하는 데이터의 경계를 구분한다
② 상대적으로 TCP 보다 전송 속도가 빠르다
③ 실시간 Streaming 서비스에 적합하다
④ 데이터그램의 전송/수신 순서가 보장된다

20. 네트워크의 OSI 7계층 중 전송 계층에 해당하는 것을 고르시오
① FTP ② HTTP ③ TCP ④ ICMP

21. 다음의 순서로 메모리에 4byte 데이터 값이 저장되어 있다. 메모리 저장 방식이 리틀 엔디안(little-endian)일 경우 실제 값은 얼마인가?( 한 칸은 1byte를 의미한다. 왼쪽 칸이 메모리의 첫 번째 바이트이다.)

| 0x01 | 0x02 | 0x03 | 0x04 |
|---|---|---|---|

① 0x01020304
② 0x02010403
③ 0x03040102
④ 0x04030201

22. 다음은 c++언어의 헤더파일에 대한 설명이다. 틀린 것은?
① 클래스나 함수의 선언부를 모아두는데 사용한다. 소스코드의 내용을 공개하지 않고 라이브러리 형태로 전달 할 때 헤더파일을 공유함으로써 어떤 함수들이 있는지 소개할 수 있다.
② 하나의 cpp 파일에 여러 개의 헤드파일을 포함하여 사용할 때는 컴파일 에러가 발생하지 않도록 종속성 순서가 중요한 경우가 있다.
③ 반드시 모든 헤더파일은 .h 확장자를 가져야 하며, 클래스나 함수의 선언 외에 정의가 포함될 경우 컴파일이 원활하지 않을 수 있다.
④ 컴파일 속도를 높이기 위해서는 헤더파일에 여러 헤더파일을 포함하기 보다는 cpp파일에서 필요한 헤더파일만 포함하는 것이 좋다.

23. C언어에서 1바이트 자료형으로 십진수 7을 이진수로 표현하면 00000111 이다. 다음 중 십진수 -7을 이진수로 표현한 것으로 맞는 것은?
① 10000111
② 11100000
③ 11111000
④ 11111001

24. 다음 프로그램을 실행하면 화면에 출력되는 값은?

```
#include <iostream>
int main()
{
    int arr[3][3] = {
        { 1, 2, 3 },
        { 4, 5, 6 },
        { 7, 8, 9 }
    };
    std::cout << arr[0][1] << arr[1][1] << arr[2][0];
    return 0;
}
```

① 257 ② 453 ③ 657 ④ 851

25. 다음 프로그램을 실행하면 화면에 출력되는 값은?

```
#include <iostream>
#include <string>

class A
{
protected:
    std::string val;
public:
    void Set(std::string v) { val = "A" + v; }
    virtual std::string Get() { return val + "A"; }
};

class B : public A
{
public:
    virtual void Set(std::string v) { val = "B" + v; }
    std::string Get() { return val + "B"; }
};

int main()
{
    B b;
    A* p = &b;
    p->Set("_");
    std::cout << p->Get();
    return 0;
}
```

① A_A ② A_B ③ B_A ④ B_B

## 2 게임 알고리즘과 설계

1. 다음은 3구간에 대한 1차 선형보간(linear interpolation)을 나타내는 수식으로 알맞은 것은?

단, 3구간은 a와 b 구간, b와 c 구간, c와 d 구간이며, 매개변수 t는 0.0~1.0의 범위를 갖는다.
t=0일 때 a값 출력, t=0.5일 때 b값 출력, t=0.75일 때 c값 출력, t=1.0일 때 d값을 출력해야 한다.

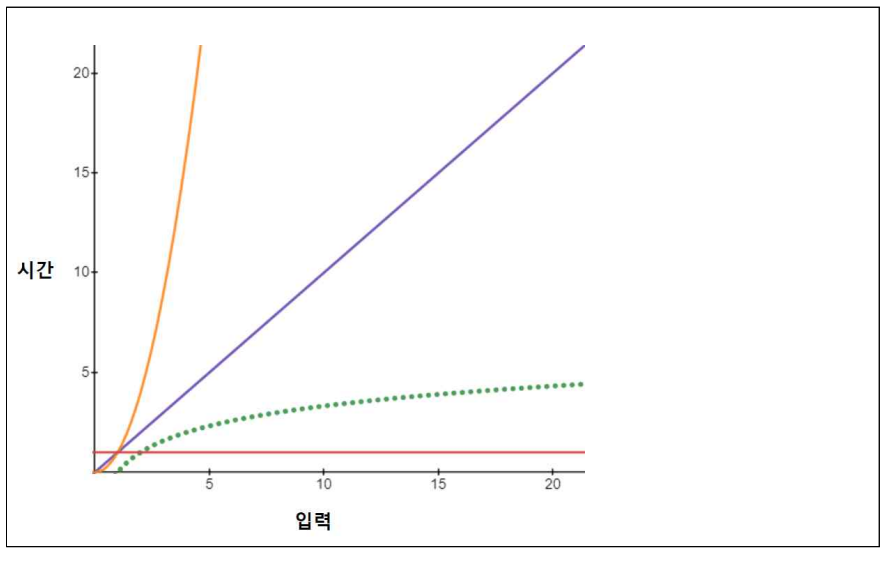
① $a(1-t)+bt$, 단 $0 \le t \le 0.5$,
$b(1-t)+ct$, 단 $0.5 \le t \le 0.75$
$c(1-t)+dt$, 단 $0.75 \le t \le 1$

② $2a(0.5-t)+2bt$, 단 $0 \le t \le 0.5$,
$2b(0.75-t)+2c(t-0.5)$, 단 $0.5 \le t \le 0.75$
$2c(1-t)+2d(t-0.75)$, 단 $0.75 \le t \le 1$

③ $2a(0.5-t)+2bt$, 단 $0 \le t \le 0.5$
$4b(0.75-t)+4c(t-0.5)$, 단 $0.5 \le t \le 0.75$
$4c(1-t)+4d(t-0.75)$, 단 $0.75 \le t \le 1$

④ $4a(0.5-t)+4bt$, 단 $0 \le t \le 0.5$,
$4b(0.75-t)+4c(0.5-t)$, 단 $0.5 \le t \le 0.75$
$4c(1-t)+4d(t-0.75)$, 단 $0.75 \le t \le 1$

2. 다음은 알고리즘의 시간 복잡도 빅오(Big O)를 비교한 그래프이다. 점선에 해당되는 것으로 알맞은 것은?



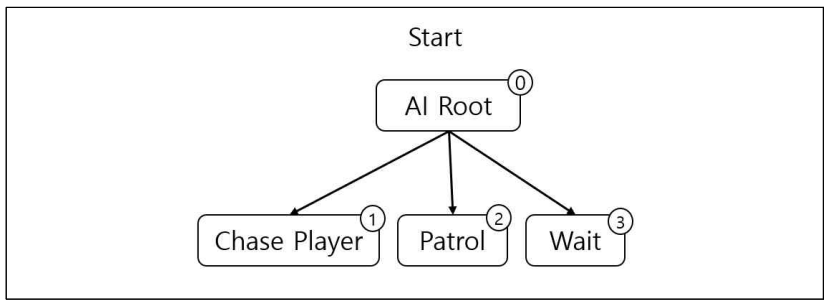

① $O(n)$ ② $O(1)$ ③ $O(\log_2 n)$ ④ $O(n^2)$

3. 다음 코드는 이중연결리스트(Double Linked List)에서 노드를 삭제하는 과정을 나타내는 코드의 일부이다. 빈칸 (A)에 알맞은 코드는?

```
struct Node
{
        int     data;
        Node *pPrev;
        Node *pNext;
};
void DeleteNode(Node *pTarget)
{
        pTarget->pPrev->pNext = pTarget->pNext;
        (       A       ) = pTarget->pPrev;
        free(pTarget);
}
```

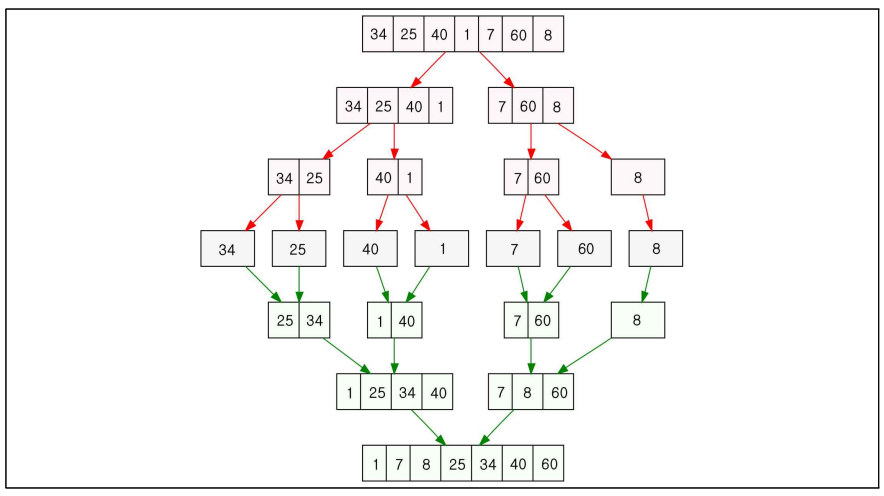
① pTarget->pPrev->pNext ② pTarget->pPrev
③ pTarget->pNext->pPrev ④ pTarget->pNext

4. 다음은 몬스터의 AI를 표현하는 행동트리(Behavior tree)이다. 깊이 우선 탐색으로 자식 노드를 수행하고 자식의 수행결과로 부모 노드의 행동을 결정한다 이를 구현하기에 알맞은 자료구조를 고르시오.



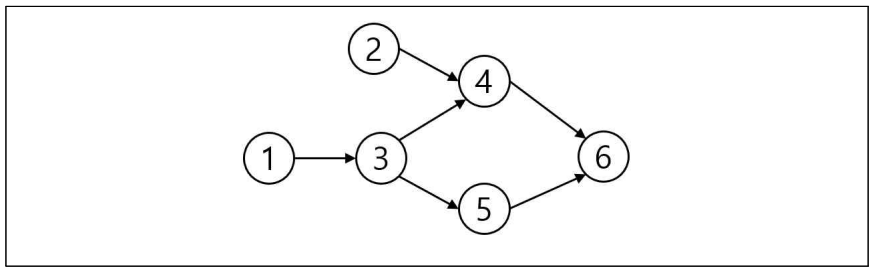

① 스택 ② 리스트 ③ 힙 ④ 큐

5. 다음 그림에 해당하는 정렬 방법을 고르시오




① 버블정렬(Bubble Sort) ② 병합정렬(Merge Sort)
③ 삽입정렬(Insertion Sort) ④ 퀵정렬(Quick Sort)

6. 다음 방향성 그래프를 보고 인접리스트로 올바르게 표현한 것을 고르시오.




① 
```
[1]->[3]
[2]->[4]
[3]->[1]->[4]->[5]
[4]->[2]->[3]->[6]
[5]->[3]->[6]
[6]->[4]->[5]
```

② 
```
[1]->[3]
[2]->[4]
[3]->[1]->[4]->[5]
[4]->[6]
[5]->[6]
[6]
```

③ 
```
[1]->[3]
[2]->[4]
[3]->[4]->[5]
[4]->[2]->[3]->[6]
[5]->[3]->[6]
[6]->[4]->[5]
```

④ 
```
[1]->[3]
[2]->[4]
[3]->[4]->[5]
[4]->[6]
[5]->[6]
[6]
```

# 4 게임프로그래밍 전문가 1-A형

**7.** B-tree에 대한 설명 중 올바르지 않은 것을 고르시오.

① Database와 같은 큰 데이터 블록을 읽고 쓰는 시스템에 적합하다

② B-tree는 이진트리의 한 유형이다.

③ 외부 파일로부터 메인 메모리로 읽는 횟수를 줄이는데 주안점이 있다.

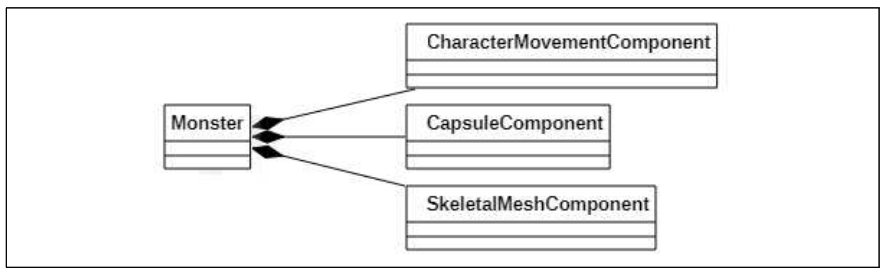
④ 하나의 노드에서 둘 이상의 데이터를 허용한다.

**8.** 다음은 어떠한 문제를 해결하기 위해 작성한 함수이다. 함수의 $O$ (Big-O) 시간 복잡도를 고르시오.

```
int function(int n,int j)
{
    int sum = 0;
    for (int i = 0; i<n; i++)
    {
        if (i>j)
            sum = sum +1;
        else
        {
            for (int k = 0; k < n; k++)
              sum = sum -1;
        }
    }
    return sum;
}
```

① $O(n)$ ② $O(1)$ ③ $O(\log_2 n)$ ④ $O(n^2)$

**9.** 다음 몬스터 클래스는 컴포넌트 기반 개발(Component Based Development)으로 설계하였다. 보기에서 설계에 대한 올바르지 않은 설명을 고르시오.




① 몬스터가 사라지면 컴포넌트도 사라진다.

② 컴포넌트 베이스드 설계방식은 클래스 사이의 의존 문제를 해결 해준다.

③ 컴포넌트 독립된 기능으로 유지보수가 간편해진다.

④ 충돌에 대한 반응은 캡슐 컴포넌트에 구현한다.

**10.** 다음은 고정 크기 배열로 구현한 자료구조로 자료를 추가, 삭제한 결과이다. 작동방식과 가장 가까운 자료구조를 고르시오.

| | | | | | |
|---|---|---|---|---|---|
| Push 14 | **14** | | | | |
| Push 22 | **14** | **22** | | | |
| Push 13 | **14** | **22** | **13** | | |
| Pop | | **22** | **13** | | |
| Pop | | | **13** | | |
| Push -6 | | | **13** | **-6** | |
| Push 9 | | | **13** | **-6** | **9** |
| Push 8 | **8** | | **13** | **-6** | **9** |

① 스택(Stack) ② 큐(Queue)

③ 순환큐(Circular Queue) ④ 연결리스트(Linked List)

**11.** 다음 설명은 자료구조에 대한 일반론을 나타낸다. 이 중에서 설명이 바른 것은?

① 스택과 큐는 배열이나 리스트처럼 데이터의 입력과 출력을 자유롭게 할 수 있어 언제든지 중간에 데이터의 삽입과 삭제를 할 수 있다.

② 함수들의 호출과정은 스택 자료구조를 가지고 있어 함수 내의 지역변수(Local Valuable)들은 메모리의 스택영역에서 만들어 진다.

③ 스택(Stack)은 LIFO(Last In First Out) 방식으로 동작하기 위하여 비선형의 그래프 자료구조를 갖는다.

④ 연결리스트(Linked List)의 검색은 재귀적인 특성을 갖고 있어 검색 복잡도는 $O(n^2)$이다.

**12.** 다음은 트리구조를 만들기 위한 노드(Node) 구조체를 나타내고 있다. 이 자료구조에 가장 적당한 것은?

```
typedef struct _bTreeNode{
  BTData data;
  struct _bTreeNode *left;
  struct _bTreeNode *right;
} BTreeNode;
```

① 이진 트리(Binary Tree) ② 쿼드 트리(Quad Tree)

③ 옥트리(Octree) ④ 연결리스트(Linked List)

**13.** 다음의 해시 테이블(Hash Table)에 대한 설명 중 잘못된 것은?

① 해시 테이블(Hash Table)은 해시 맵(Hash Map)이라고도 부르며 평균적으로 삽입, 검색, 삭제 작업에서 시간복잡도가 O(1)이다.

② 해시 함수(Hash Function)는 인자에 키(Key)를 넣어 데이터를 고정된 길이의 해시 값(Hash Value)으로 리턴해주는 함수이다.

③ 해시테이블 시간 복잡도는 평균 O(1)이고, 해시 값의 중복이 많은 최악의 경우에는 연결된 리스트들까지 검색을 해야 하므로 O(n)까지 증가한다.

④ 해시 함수를 구하는 시간이 모든 데이터를 비교하는 시간보다 느리다면, 데이터가 적을 경우 성능이 굉장히 좋아진다.

**14.** 다음은 정렬된 데이터에 대한 이진검색에 대한 C++ 코드이다. 빈칸 (A)에 알맞은 코드는?

```
#include <iostream>
using namespace std;

int binarySearch(int arr[], int l, int r, int x){
    int mid;
    int ret_val = -1;
    if( r >= l ){
        mid = l + (r - l) / 2;
        if( arr[mid] == x )
            { ret_val = mid; }
        else if( arr[mid] > x )
            {                ( A )                    }
         else
            { ret_val = binarySearch(arr, mid + 1, r, x); }
    }
    return ret_val;
}

int main(void){
    int arr[] = { 3, 6, 9, 11, 15, 17, 21, 25 };
    int x = 11;
    int n = sizeof(arr) / sizeof(arr[0]);
    int result = binarySearch(arr, 0, n - 1, x);
    if( result == -1 )
        { cout << x <<" is not in array" << endl; }
    else{ cout << x <<" is in array at " << result << "-th." << endl;
}
    return 0;
}

>> (결과)
11 is in array at 3-th.
```

① ret_val = binarySearch(arr, l, arr[mid] - 1, x);

② ret_val = binarySearch(arr, l, mid - 1, x);

③ ret_val = binarySearch(arr, arr[mid] + 1, r, x);

④ ret_val = binarySearch(arr, 1 - arr[mid], r, x);

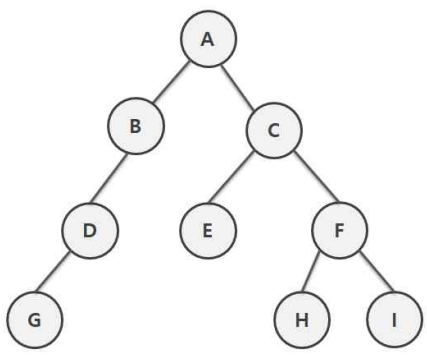
15. 다음의 균형 탐색 트리(Balanced Search Tree)에 대한 설명 중에서 틀린 것은?

① 균형 탐색 트리가 제안된 주된 목적은 메모리 저장을 효율적으로 하기 위해서다.
② 이진탐색 트리의 탐색 연산을 변경 없이 사용할 수 있다.
③ 균형 탐색 트리에는 AVL, Red-Black, 2-3 Tree 등이 있다
④ AVL 트리의 성질을 만족하는 지는 루트 노드를 기준으로 가장 긴 왼쪽 경로와 가장 긴 오른쪽 경로 길이의 차이인 균형인수(balance factor) 값을 찾아 확인한다.

16. 다음의 그래프(Graph)에 대한 설명 중 바른 것은?

① 그래프에서 간선 가중치의 합이 가장 큰 신장트리를 최소신장트리(Minimum Spanning Tree)라고 한다.
② 가중치 그래프는 처음과 끝의 정점이 주어진 순환이 있는 그래프이다.
③ 프림 알고리즘은 두 개의 노드들의 집합을 합칠 때마다 두 집합을 연결하는 하나의 간선이 정해지므로 간선을 하나씩 더하는 것이다.
④ 내비게이션 그래프(Navigation Graph)는 게임 내 에이전트(Agent)가 방문할 수 있는 모든 장소와 그 지점들을 연결한 것이다.

17. 다음 그림과 같은 이진트리에서 전위 순회(Pre-Order Traversal)를 통해 얻은 순서에 맞는 것은?




① A B C D E F G H I
② A C F I H E B D G
③ A B D G C E F H I
④ G D B E A H C F I

18. 다음 코드는 정렬 알고리즘을 구현하는 함수의 일부분이다. 가장 적절한 알고리즘으로 알맞은 것은?

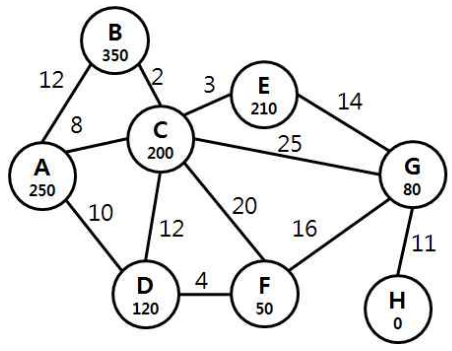
```
// Divide
void _Sort(int arr[], int left, int right) {
    if(left==right) return;
    int mid;
    mid = (left + right) / 2;
    _Sort(arr, left, mid);
    _Sort(arr, mid + 1, right);
    _Task(arr, left, right);
}

// *****
void _Task(int arr[], int left, int right) {
    int L, R, k, a;
    int mid = ( left + right ) / 2;
    L = left;
    R = mid + 1;
    k = left;

    while (L <= mid && R <= right)
        tmp[k++] = arr[L] <= arr[R] ? arr[L++] : arr[R++];
    if (L>mid)
        for (a = R; a <= right; a++) tmp[k++] = arr[a];
    else
        for (a = L; a <= mid; a++) tmp[k++] = arr[a];
    for (a = left; a <= right; a++) arr[a] = tmp[a];
}
```

① 선택정렬(Selection Sort)
② 버블정렬(Bubble Sort)
③ 병합정렬(Merge Sort)
④ 삽입정렬(Insert Sort)

19. 다음과 같은 그래프(Graph) 에서 A* 알고리즘을 적용하여 출발지점(A) 에서 목표지점(H)까지 최단거리 탐색을 하려 한다. 이때 지점 (F) 에서의 평가 값으로 알맞은 것은?




[ 범례 ]

A 100 A - 지점(Vertex)의 이름, 100 - 지점 A에서 목표 H 까지 상대거리인 휴리스틱 값

A 30 B 30 - 지점 A와 B 사이의 경로 비용

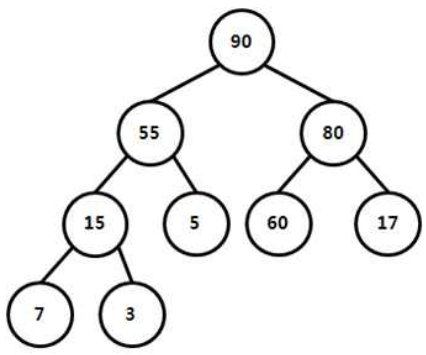
① 64
② 78
③ 88
④ 94

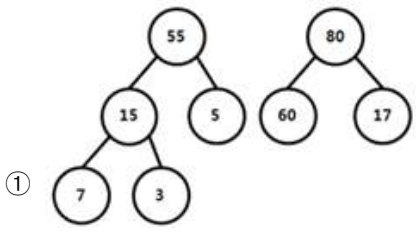
20. 다음의 설명은 문자열 매칭 알고리즘에 대한 것이다. 빈칸 (A)에 알맞은 알고리즘은 어떤 것인가?

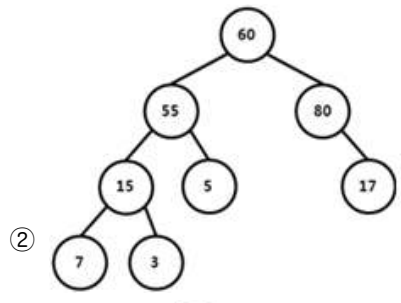
- ( A ) 알고리즘은 문자열의 앞보다는 뒤에서 불일치가 일어날 확률이 높다는 성질에 기반하고 있다.
- 검색 문자열의 오른쪽 끝부터 왼쪽으로 진행하며 비교한다. 일치하지 않는 문자가 나타나면 정해진 규칙에 따라 오른쪽으로 스킵(skip)해서 다시 진행한다.
- 불일치 패턴 발견 시 오른쪽으로 얼만큼 이동할지를 미리 테이블로 작성해놓는데 이 테이블을 스킵 테이블(skip table)이라고 한다.

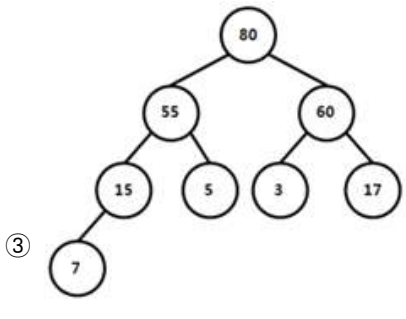
① 브루트-포스(Brute-Force) 알고리즘
② 라빈-카프(Rabin-Karp) 알고리즘
③ KMP(Knuth-Morris-Pratt) 알고리즘
④ 보이어-무어(Boyer-Moore) 알고리즘

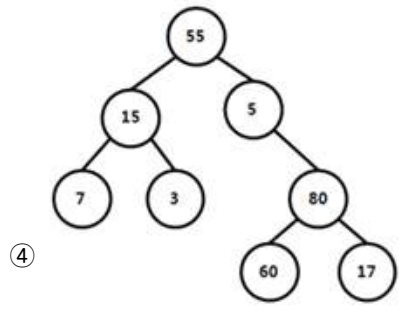
21. 다음과 같은 힙(Heap)에서 최상위 노드 삭제 시 결과로 바르게 표현한 것은?




① 


② 


③ 

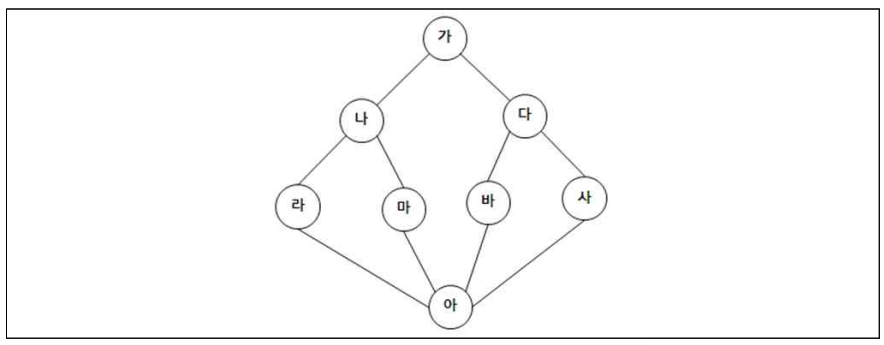

④ 

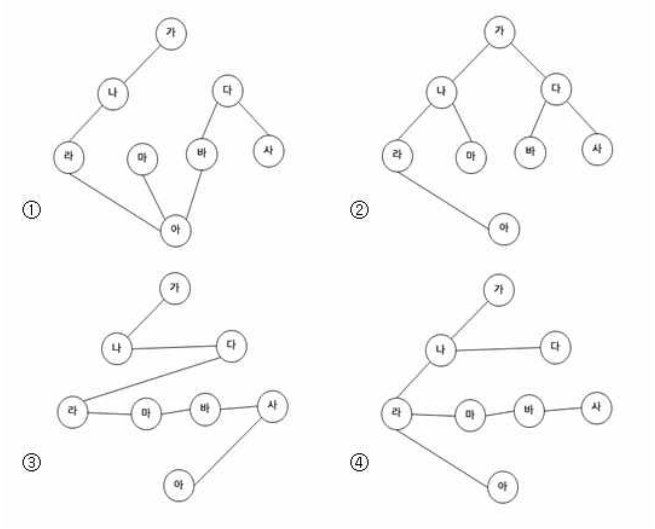
22. 다음 데이터에 대하여 이진탐색을 한다고 하자. 48을 찾기 위한 비교 횟수와 26을 찾기 위한 비교 횟수는?

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 5 | 11 | 19 | 26 | 48 | 59 |

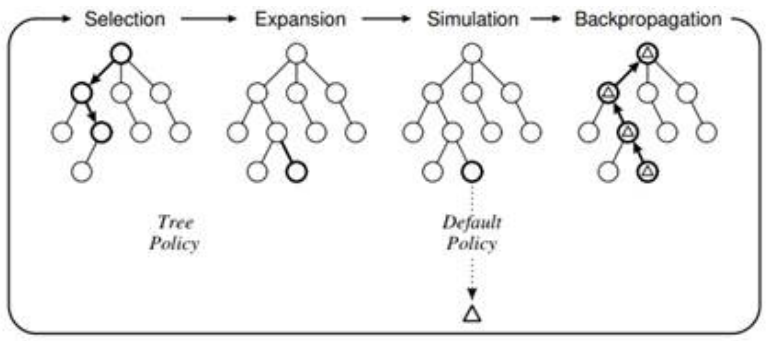
① 2회 - 3회 ② 6회 - 3회 ③ 6회 - 2회 ④ 3회 - 2회

23. 다음 그림과 같은 그래프가 주어졌다고 가정했을 때, 이를 "가"노드를 시작점으로 깊이 우선 검색(DFS)을 실시하였다. DFS에 대한 그림 설명으로 가장 적당한 것은? (순서는 가나다 순 임)




24. 다음은 몬테카를로 트리탐색(Monte Carlo Tree Search, MCTS)에 대한 설명이다. 이중에서 틀린 것을 고르시오.




① 바둑, 장기, 보드게임, 알파고에서 사용되며 시뮬레이션을 통해 승률이 낮은 행동을 찾아 실패를 줄이도록 하는 알고리즘이다.
② 몬테카를로 트리 탐색은 선택, 확장, 시뮬레이션, 역전달의 4단계 과정을 거치면서 순회하는 방법이다.
③ 한 게임이 끝까지 진행되어 그 게임의 결과가 아직 정해지지 않았지만 끝까지 갔을 때 정확한 그 결과가 나오는 하나의 게임을 의미하는 플레이아웃(Playout)을 기초로 무작위선택 적용하는 방법이다.
④ 최대 최소값이 존재하고, 게임 규칙이 존재해야 하고, 게임 길이가 제한되어 있어야 한다는 3가지 조건이 만족되어야 사용가능한 것으로 알려져 있다.

25. 다음은 C 언어로 작성된 삽입정렬 알고리즘이다. 빈 칸 [A]에 알맞은 것은?

```
void algorithm(int *data, int n) {
        int i,j,m;
        for( i = 1; i < n ; i++ ) {
                m = data[i]; j = i;
                while( --j >= 0 && m < data[j] )
                { [   A   ] }
                data[j+1] = m;
        }
}
```

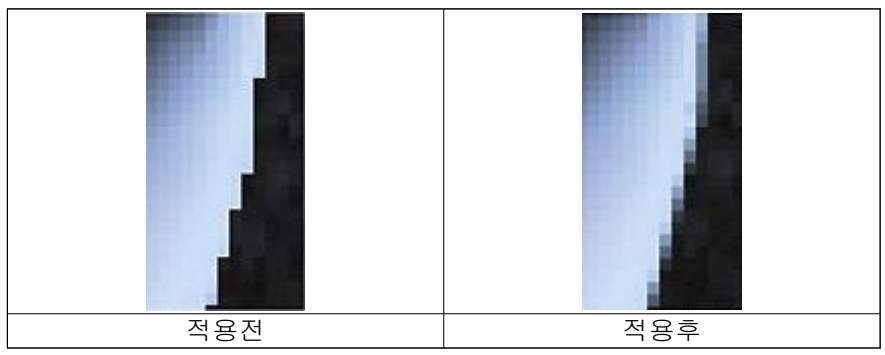
① data[j] = data[j+1]; ② data[j+1] = data[j];
③ m = data[j]; ④ m = data[j+1];

## 3 게임 콘텐츠 프로그래밍

1. 다음 그림처럼 Color LUT(Look Up Table)를 사용하여 게임화면의 전체적인 색상을 조작하는 포스트 프로세싱 효과를 무엇이라 하는가?


① Morph ② Blur
③ Color Correction 혹은 Color Grading ④ Light Map

2. 다음 그림처럼 이미지의 경계면을 부드럽게 처리하는 기법을 무엇이라 하는가?



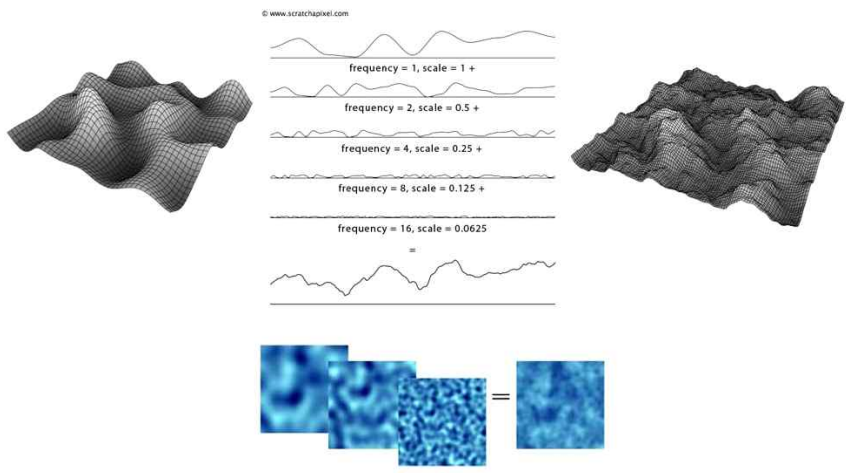

① Anti Aliasing ② Brightness
③ SSAO ④ HDR

3. 다음 그림처럼 초점거리에 맞춰서 전방이나 후방을 Blur처리하여 캐릭터를 부각시켜 시선을 집중하도록 연출하는 기법을 무엇이라고 하는가?

① Contrast ② Ray Tracing
③ DOF(Depth Of Field) ④ Tone Mapping

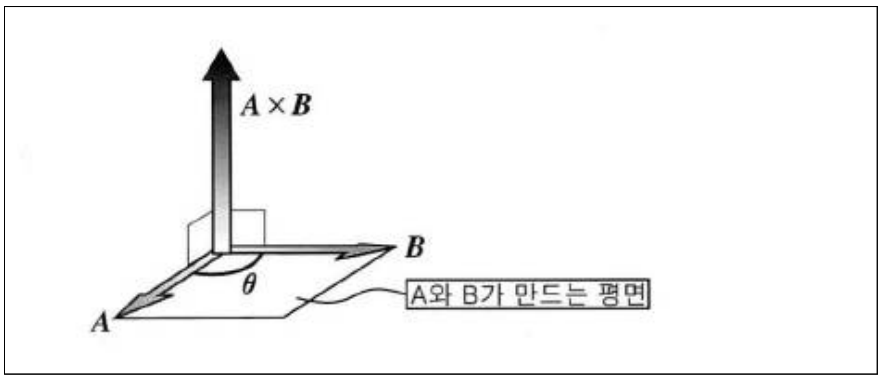
4. 다음은 무엇에 대한 설명인가?

현실과 가까운 구름이나 지형을 처리하기 위해서 사용하는 기법으로, 여러 개의 노이즈(noise)를 섞어서 만드는 것이 특징이다.



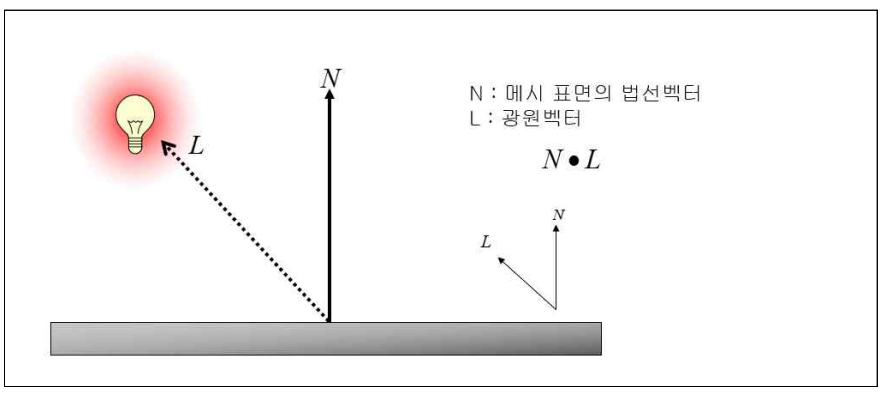

① Scale ② Frequency
③ Amplitude ④ Perlin Noise

5. 다음 그림처럼 두 개의 벡터를 연산하여 수직인 벡터를 만들어 내는 연산을 무엇이라고 하는가?



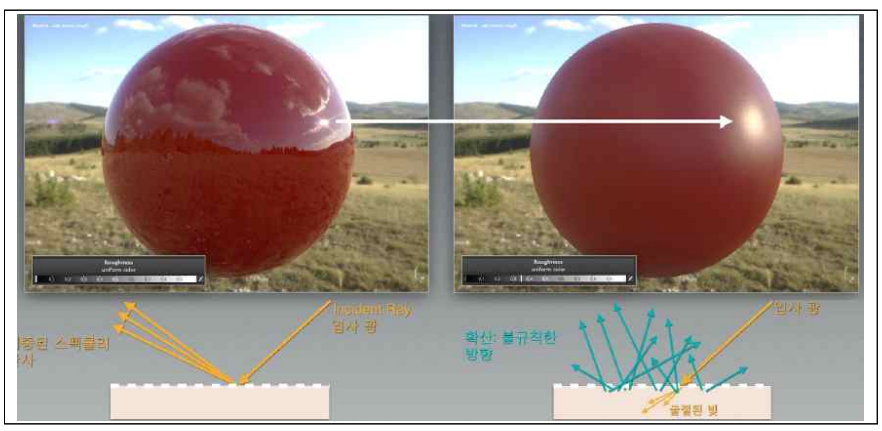

① 벡터 덧셈
② 벡터 정규화
③ 벡터 내적
④ 벡터 외적

6. 다음 그림에 보이는 두 벡터의 연산으로 확산광(Diffuse)의 값을 구하는데, 이때 사용하는 벡터 연산의 명칭은 무엇인가?




① 내적
② 외적
③ 정규화
④ 덧셈

7. 다음 그림처럼 표면의 거친 정도(roughness)를 고려하여 반사광을 물리기반으로 계산하는 방식을 무엇이라 하는가?

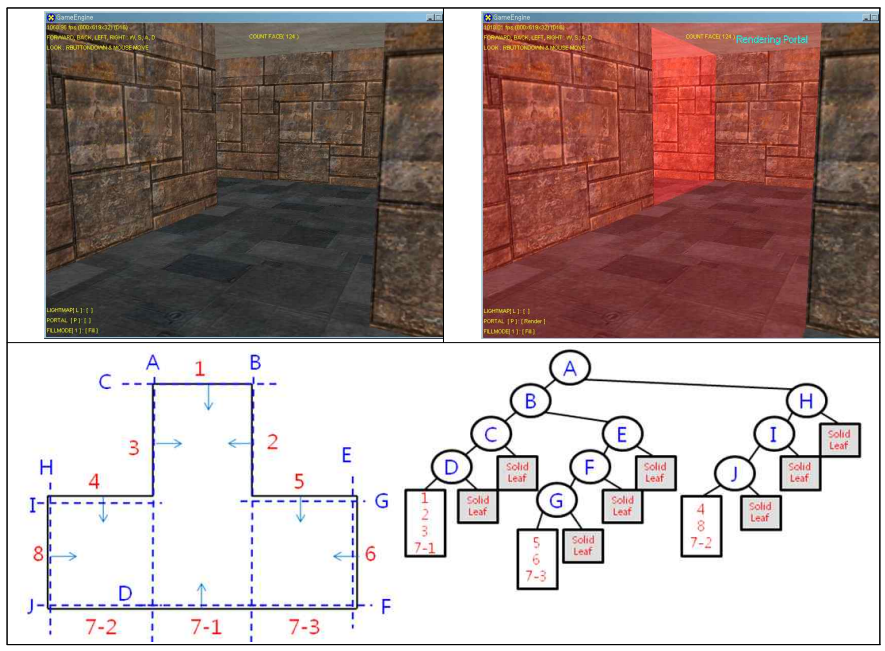



① Phong Shading
② Stylized Shading
③ Physically Based Shading
④ Flat Shading

8. 실제로는 3차원 월드에 다수의 오브젝트가 존재하지만 카메라에서 보이는 영역은 한정되어 있다. 따라서 카메라에서 보이는 영역만을 렌더링하고 보이지 않는 영역을 제외시키는 기법을 무엇이라고 하는가?

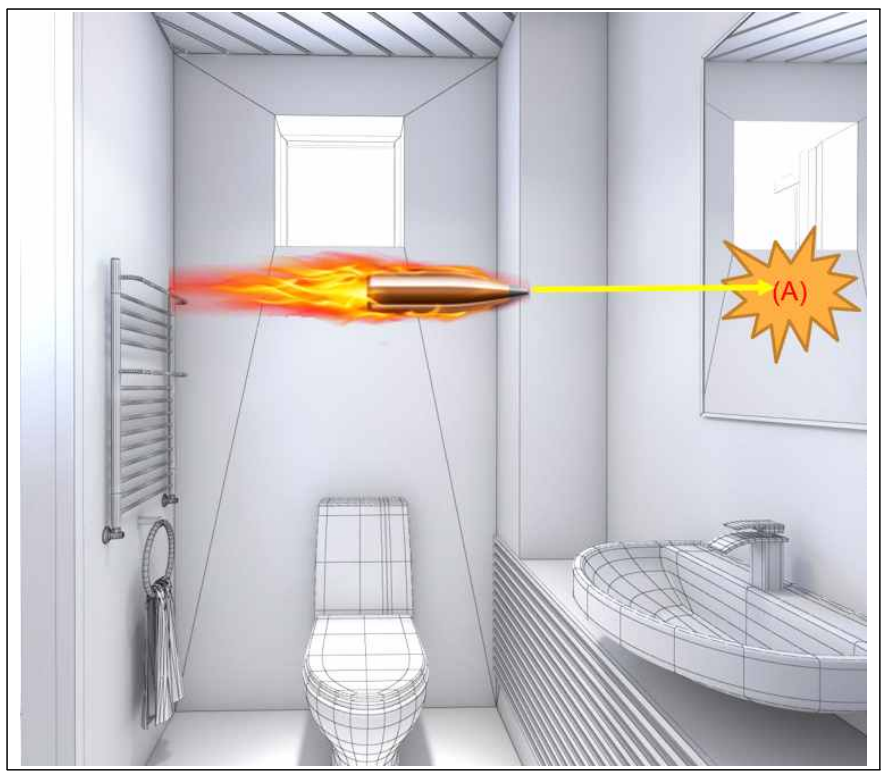
① 후면 컬링
② 절두체 컬링
③ 상하체 분리
④ 키프레임 애니메이션

9. 다음 그림처럼 공간을 특정 면을 기준으로 앞과 뒤로 분할하여 트리 구조 형태로 처리하는 기법을 무엇이라고 하는가?




① BSP Tree
② Kd Tree
③ Quad Tree
④ Octree

10. FPS 게임에서 다음 그림처럼 총알이 벽에 맞았을 경우 피탄 부위(A)를 찾기 위해서 우리는 직선과 평면의 교점을 구할 필요가 있다. 이때 평면의 방정식이 사용되는데, 원점으로부터 d만큼 떨어져 있고, 평면의 법선벡터가 (a,b,c)라 할때 평면의 방정식을 구하시오

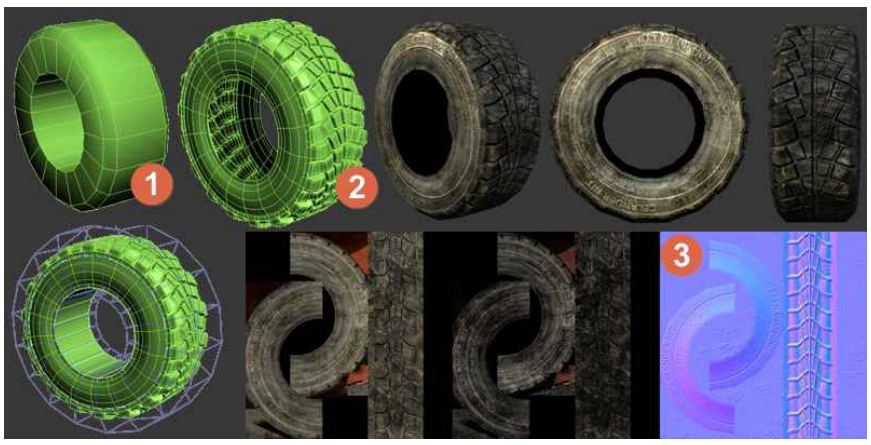



① $cx+by+az+d=0$
② $ax+by+cz-d=1$
③ $ax+by+cz+d=0$
④ $cx+by+az+d=1$

11. 3차원 그래픽 처리와 관련한 아래 설명 중 올바르지 않은 것은?

① 각 픽셀에서 조명식을 계산하는 것이 가능하다.
② OpenGL은 저수준의 렌더링을 지원하는 3차원 라이브러리다.
③ 알파 블렌딩은 픽셀을 출력할 때 투명 또는 반투명하게 출력하는데 사용하는 방법이다.
④ 같은 재질이나 텍스쳐를 사용하는 오브젝트들을 묶어서 처리하지 않고 각각 따로 처리하는 것이 렌더링 속도가 빠르다.

12. 아래 그림에서 실제 게임 플레이에서는 아래 그림의 1번과 같은 로우폴리곤의 타이어 오브젝트를 사용하지만 실제와 같은 정교한 렌더링을 위해서 2번과 같은 하이폴리곤을 사용하여 3번의 텍스쳐 이미지를 제작한다. 아래 보기 중 3번과 같은 텍스쳐 이미지에 해당하는 것으로 알맞은 것은?

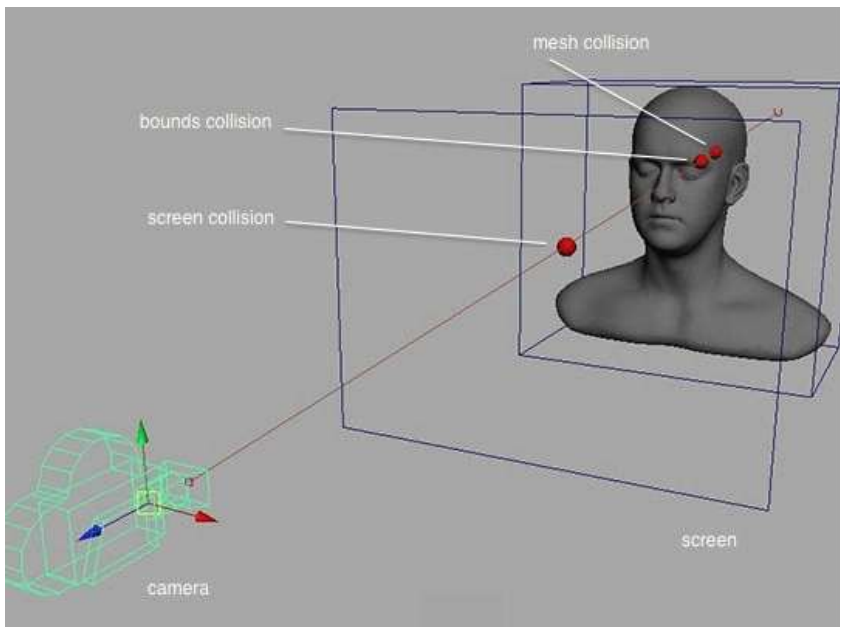



① 큐브맵(cube map)　② 빌보드(billboard)
③ 법선맵(normal map)　④ 글로우맵(glow map)

13. 아래 보기에서 3차원 그래픽 처리에서 법선 벡터의 사용에 대한 설명 중 올바르지 않은 것은?

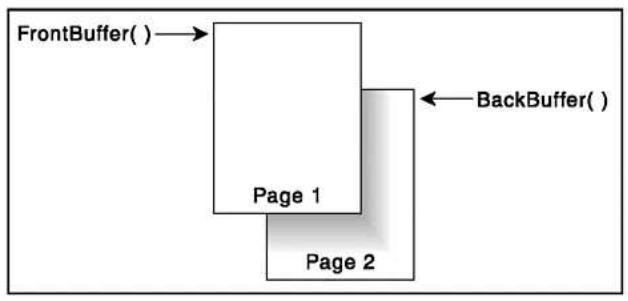
① 법선 벡터를 이용하면 렌더링할 필요가 없는 폴리곤을 가려 낼 수 있다.
② 주어진 평면의 법선 벡터는 평면 위의 두 벡터의 내적을 통해서 구할 수 있다.
③ 법선맵(Normal Map)은 정점의 법선 벡터값을 이미지 형태로 저장한 것이다.
④ 3차원 렌더링의 조명 계산에 사용한다.

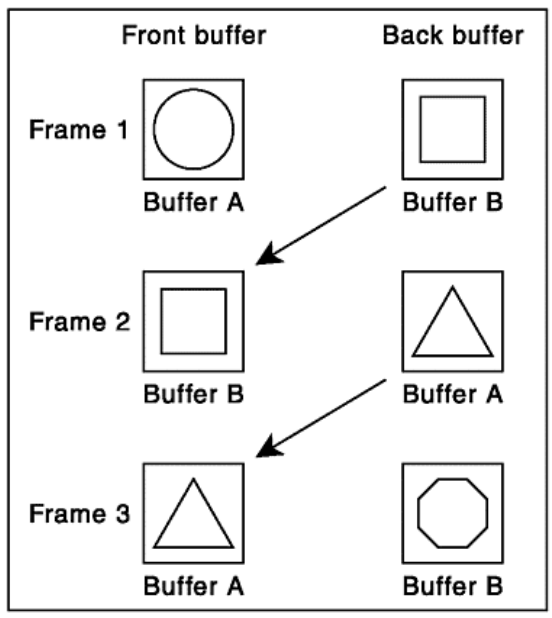
14. 아래 그림은 마우스를 이용하여 3차원 공간 내의 물체를 선택하는 과정을 나타낸 것이다. 이와 관련한 보기 설명 중 올바르지 않은 것은?




① 카메라 위치와 마우스 좌표를 이은 직선(ray)를 구해서 사용한다.
② ①번의 ray 계산에는 3차원 렌더링 파이프라인의 변환 중 월드 변환에 사용하는 월드 행렬을 사용한다.
③ ①번에서 구한 ray와 3차원 공간내 물체들의 바운딩 볼륨(bounding volume)과의 충돌을 계산한다.
④ ③번 과정에서 자신의 바운딩 볼륨과 충돌한 메쉬 오브젝트의 각각의 폴리곤에 대한 충돌 검사를 한다. 이 계산은 ③번에서 사용한 ray-bounding 충돌 계산보다 시간을 더 소요하는 복잡한 계산이다.

15. 게임 엔진 내부에서 렌더링 처리시 아래 그림과 같이 전면버퍼(Front Buffer)와 후면 버퍼(Back Buffer)의 두 가지 버퍼를 생성한 다음, 렌더링시 사용한다. 후면버퍼와 전면버퍼를 바꿔주고(swap), 전면버퍼의 이미지는 화면에 렌더링되어 나타난다. 이러한 처리를 플리핑(Flipping)이라고 한다. 아래 보기 중 후면 버퍼와 전면 버퍼로 구분하여 플리핑(Flipping)을 이용해서 렌더링하는 이유로 알맞은 것은?




① 애니메이션시 화면이 깜박이지 않게 하기 위해서이다.
② 후면 버퍼는 3D 오브젝트의 LOD(Level Of Detail) 처리에 사용된다.
③ 전면 버퍼의 각 픽셀에서 조명을 계산하기 위해서 두 개의 버퍼가 필요하다.
④ 전면 버퍼는 정점 셰이더 계산용, 후면 버퍼는 픽셀 셰이더 계산용으로 사용한다.

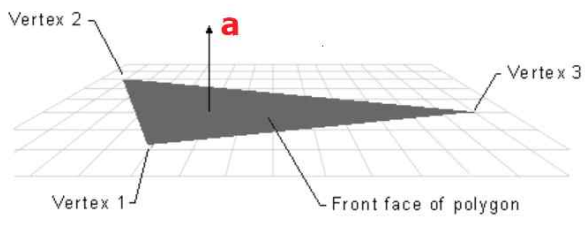
16. 그림과 같은 1인칭 슈팅 게임의 플레이어 캐릭터를 처리하고자 한다. 카메라는 캐릭터의 눈높이에 위치하고, 화면의 라이플 모델은 플레이어 캐릭터의 자식 모델로 추가되어 있다. 구현과 관련한 아래 보기의 설명 중 알맞지 않은 것은?

① 키보드로 캐릭터를 움직일 때 카메라 행렬의 위치 값을 변경한다.
② 마우스로 캐릭터의 시선을 변경하는 것은 카메라 행렬의 회전이 필요한데, 이 계산은 사원수(quaternion)를 사용해서 처리할 수도 있다.
③ 캐릭터와 지면의 충돌 계산은 캐릭터를 포함하는 실린더를 사용해서 처리할 수 있다.
④ 라이플의 세계좌표(world position)의 위치(position)값은 플레이어 캐릭터의 지역좌표계(local coordinate system)의 위치값에 라이플의 지역좌표계의 위치값을 더해서 계산할 수 있다.

17. 3차원 게임에서 아래 그림과 같은 폭발효과를 제작하려고 한다. 이 폭발효과는 시점에 따라 회전되지 않고 x, y, z 어느 방향에서나 관찰이 가능해야 한다. 이에 대한 처리로 아래 보기 중 가장 알맞은 것은?

① 파티클 (Particle)
② 포그 (Fog)
③ 역기구학(Inverse Kinematics)
④ 엠비언트 오클루젼(Ambient Occlusion)

18. 아래 그림의 화살표 a는 폴리곤에 수직하는 벡터며, 회색 부분은 폴리곤의 앞면(front)을 나타낸다. 아래 보기 중 이 벡터에 해당하는 것으로 알맞은 것은?




① 탄젠트 벡터 ② 법선 벡터
③ 시선 벡터 ④ 정점 벡터

19. 오픈 월드 게임에서 플레이어가 차량을 운전할 때 특정 구간을 정해진 속도 이상으로 과속하면 AI로 움직이는 경찰차가 플레이어를 쫓아가도록 만들려고 한다. 플레이어 차량의 구간 평균 속도를 이용하여 이 기능을 구현하려고 할 때, 아래 보기 중 평균 속도로 알맞은 것은? 단, 아래 그림의 왼쪽 과속카메라는 A지점에서의 차량 속도를 측정하고, 오른쪽 과속 카메라는 B지점에서의 차량 속도를 측정한다.

**단, 차량은 A에서 B방향으로 전진 중이다.**
a: A지점에서 측정한 시간
b: B지점에서 측정한 시간
d: A와 B 구간의 거리


① 평균속도 = d / (a-b) ② 평균속도 = d / (b-a)
③ 평균속도 = d / (a+b) ④ 평균속도 = d / (a*b)

20. 3차원 게임에서 텍스쳐 매핑은 가상의 3차원 물체의 표면에 세부적인 질감의 묘사를 하거나 색을 칠하는 기법이다. 아래 보기의 텍스쳐 매핑과 관련한 설명 중 바르지 않은 것은?

① 환경매핑(environment mapping)을 사용하면 스포츠카의 표면에 주변 빌딩과 조명이 반사되는 효과를 연출할 수 있다.
② 텍스쳐 매핑은 오브젝트의 표면에 여러 번 반복해서 나타나도록 매핑할 수도 있다.
③ 밉맵(mipmap)은 렌더링 속도를 향상시키기 위한 목적으로 기본 텍스처와 이를 연속적으로 미리 축소시킨 텍스처들로 이루어진 비트맵 이미지의 집합이다.
④ 폴리곤에 텍스쳐 이미지를 매핑하기 위해서 텍스쳐 이미지의 (u, v) 좌표가 필요하며 이들 좌표는 텍스쳐 이미지에 함께 저장된다.

21. 3차원 게임 캐릭터 애니메이션과 관련한 설명 중 바르지 않는 것은?

① 두 개의 애니메이션을 혼합하여 출력하는 것이 가능하다.
② 전체 애니메이션 구간 중 중요한 구간의 프레임들에 애니메이션 키 값을 등록하고 나머지 키 사이의 값들은 보간을 통해서 생성한다.
③ 캐릭터의 관절은 행렬 연산을 통해서 처리가 가능하다.
④ 캐릭터 메시 데이터의 정점은 정점셰이더를 통해 계산하지만 정점의 색상 변경은 픽셀 셰이더를 사용해야만 한다.

22. 스키닝(Skinning)이란 계층적 애니메이션과 뼈대 애니메이션방식의 단점인 관절부위의 문제를 해결하기 위한 방식의 애니메이션이다. 스키닝을 사용한 애니메이션에서는 메시를 구성하는 각각의 정점이 뼈대로부터 얼마만큼의 영향을 받는가를 가중치 값으로 설명하는데 일반적으로 이 가중치 값의 형식(type)으로 알맞은 것은?

① char ② short ③ float ④ std::string

23. 카메라 변환과 투영 변환을 거친 정점의 위치로 알맞은 것은?

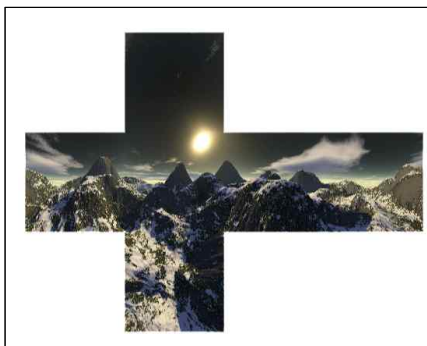
① (-1, -1, 0) 과 (1, 1, 1) 사이 ② (-1, -1,-1) 과 (1, 1, 1) 사이
③ (-1, -1,-1) 과 (1, 1, 0) 사이 ④ (-1, -1, 0) 과 (1, 1, 0) 사이

24. 게임 개발에는 다양한 수학적 계산이 요구된다. 게임 개발과 관련한 수학적 계산 중 미분 계산이 필요한 것으로 알맞은 것은?

① 카메라(시선 벡터)와 광원 벡터가 이루는 각
② 폴리곤 모델의 삼각형의 면적 계산
③ 임의의 힘이 가해진 물체의 운동 재현
④ 충돌 판정을 위한 평면 방정식의 법선 벡터

25. 다음은 픽셀 셰이더에서 시선벡터의 반사벡터를 구하여 환경맵핑을 구현한 것이다. 이때 사용된 텍스처의 종류로 적합한 것은 무엇인가?

```
float4 ps_main( PS_INPUT Input ) : COLOR0
{
    float3 N = normalize(Input.Normal);
    float3 I = normalize(Input.ViewVec);
    float3 R = reflect(-I, N);
    float4 ret = texCUBE(baseMap, R);
    return ret;
}
```

| 사용된 텍스처 | 출력화면 |
|---|---|
|  |  |

① volume texture ② cube map texture
③ normal map texture ④ occlusion texture